Hobby engine to game engine

Aras Pranckevičius
Unity Technologies
@aras_p
http://aras-p.info
/me

- At Unity since 2006, graphics plumbing
- Demoscene 2003-2005, own engine
- Small games and Kinect-wannabe startup 2001-2004, in-house engine
- Hobby coding before that
Fair question:

• Look at this awesome thing one person did in his/her spare time!

• Why can’t real engines with many paid programmers do it ?!?!?
TL;DR answer:

- Because they are solving different problems
- With different constraints

- Well, that’s it. Questions?
Story 1
What are 100 programmers doing?
Graphics in bigger picture

• For many, “engine” means graphics first for some reason

• But: R&D staff at Unity: ~100

• Working directly on rendering: ~10

• What do the others do?
Editor & Tools

• Need UI for everything

• Workflows, UX, consistency, simplicity, discoverability

• Massive problems, especially as you gain more & more features!
Scripting

• Scripting runtimes (Mono/il2cpp/.NET)
• Garbage Collection tuning, R&D
• Engine script bindings
• Debugging (MonoDevelop fork/addins)
• Script editing (MonoDevelop fork/addins)
• Script API design / breakage / upgrade
Platforms

• At least 2 people per platform, doing just “boring work”
  • New Xcode broke foo, new Android NDK broke bar, Win10 breaks baz, new iPhone needs news splash screens, Android can do TVs now, WinPhone8.1 is totally different from WP8.0 now, Apple started rejecting apps that use API foo, …
  • Consoles have endless lists of certification requirements to meet.
Core

- Job schedulers, memory allocators, profiling, asset loading & streaming, logging, base platform stuff
- Entity/component systems
- Input
- Video playback
- ...


Other things!

• AI (navigation, crowds, behavior trees, …)

• Networking

• Audio (ties a lot into tools/UI)

• Animation (runtime, IK, retargeting, state machines, UI)

• 2D (runtime part easy; workflows/UI hard)

• In-game UI
Build engineering & infrastructure & content

- Build systems
- Version control
- Build/test farm
- IDE configs, etc.
- Documentation (APIs, manuals, references)
- Example code / projects / packages
- Translations
QA

- And I didn’t even talk about QA yet!

- It’s a world unto itself: engineers, internal QA tools, reporting systems, crash analysis, coverage, manual testing, sifting through incoming reports, …
All that adds up

• Somehow, that adds up to a lot of people :)
Working in a team

• Need to communicate!
  • Keep others updated on relevant things
  • Avoid duplicated work
  • Potentially clashing features
  • Do work in a consistent way
People are different

• Will be someone who’s way better than you
  • Can be inspiring
  • Or intimidating / depressing

• Will be someone who’s not as good as you
  • Can be promising
  • Or annoying / demotivating
Different cultures

• Programmers and non-programmers

• Local culture differences
Story 2
Environment & infrastructure
Work environment

• Super-distributed, flexible hours, often work from home

• Skype, emails, wiki, twitter :)}
### Done, possibly needs more tests:

- Reflection Probes: iterative baking
- Reflection Probes: polish, bugfix
- Universal shader: use it as default one for new materials / imports; make old shaders legacy
- Remove Shader Fog Patching (for Metal/consoles)
- Shader Loading & Hiccup Optimization

### Critical to ship 5.0:

- Directional lightmaps encoding
- Ubershader Mobile: optimize for mobiles
- Physically Based Surface Shaders
- Deferred shading: configurable g-buffer layout
- Shaders: strip unused lightmap variants at build-time (now we have lots due to Enlighten/DirLM)
- Fix Gamma/Linear lighting/shading discrepancies, remove x2
- Document everything above

### Not critical for 5.0:

- Binary Shader Serialization
- Deferred shading: better support for surface shaders
- Legacy shaders: return constant alpha
To Do

DX11 VP matrix issue

Point shadows: render into depth cubemap! (from [unread message])

Remove dyn batching

Material params reordered on serialize due to FastPropName

Shader.SetGlobalDefaultTexture, so that is fetches when material has null.

DX11, some funky crash in texture load on image fx tester project?!

[unread message]

[unread message]

Doing

Shader compiler hang on 5.0 issue

Deferred/Surface

[due date: 0/6]

demos/dev bugfixing

[due date: 0/2]

Command Buffers

[due date: 0/5]

Shader loading optimization

[due date: 1/0/6]

RenderSettings / Ambient

[due date: 0/2]

Misc Fixes

[due date: 0/1]

Fog patching

[due date: 0/4]

Done

Reviews: 5.0/editor, ios/metal, graphics/dev, graphics/shadow-improvements, 5.0/core

Add a card...
Version control

• Mercurial. Before: Subversion. Before: CVS.
  • Using “largefiles” extension

• 7.3GB .hg/store; 3GB - 25GB .hg/largefiles; 20-90GB needed for work

• 175000 commits

• Kallithea (fork of Rhodecode) for repo browsing/mgmt

• Some magic to strip NDA parts for source customers
ferou-2:~/unity(graphics aras$ hg branches
demos/dev 163029:f94807324bcc
trunk 163028:642dd35406a4
scripting/il2cpp/staging 162407:6a4ea27800a9
graphics/gi 162400:56ca150c4dbf
graphics/deferred-shading 162124:612507ce6169
license/dev 160009:9cb96f6524e6
release/4.5/shader-fixes 159818:235674e2ad90
graphics/gi/chamber 155453:7fe82e3cac36
core/assetbundle-improvement 155345:c4edd4a65623
demos/frame-debugger 153065:2fc8d5c8d972
partner/ 152408:eadde72bd663
core/projectupgrade 146459:d2a3db6d6230
scripting/webgl 145607:8685bccc3c4d
demos/unishader-mobile 144770:0894376cb847
scripting/il2cpp/icalls 142689:e6db2d6e3e21
core/streaming 142387:1939937c3313
vincent/assetbundle-improvement/automation 137341:37d180b799fb
doc/ 137269:dd1db70e84dd
docs 137268:238c3aa7b9a4
core/multi-scene-editing 137265:70a8e3189e14
scripting/il2cpp/refcounting 137234:b6cf3f4bab24
release/4.5/ 137209:88126cb635a9
documentation/pom.xml 137200:e69bbd3d15751
<table>
<thead>
<tr>
<th>Revision</th>
<th>Time Ago</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>23c3f37e5d0e</td>
<td>9 hours ago</td>
<td>Added support for generating UserOverride.jam in build.pl</td>
</tr>
<tr>
<td>1abee228f263</td>
<td>9 hours ago</td>
<td>Merge 5.0/editor/staging into trunk</td>
</tr>
<tr>
<td>e9000d0be372</td>
<td>9 hours ago</td>
<td>[UI] Fix regression where Canvas scaleFactor and referencePixelsPerUnit would be reset on recor</td>
</tr>
<tr>
<td>34f064f5147f</td>
<td>9 hours ago</td>
<td>merging unet/dev to unet/staging</td>
</tr>
<tr>
<td>6343117be17</td>
<td>9 hours ago</td>
<td>Added tag 5.0beta8 for changeset d779158c91f2</td>
</tr>
<tr>
<td>edc80ed5ccf0</td>
<td>9 hours ago</td>
<td>Fix SortLibraryRepresentations, a wrong comparator could crash std::sort since std::sort doesn't</td>
</tr>
<tr>
<td>ce36ae328419</td>
<td>9 hours ago</td>
<td>Merge graphics/gi/multithreaded--enlighten into trunk</td>
</tr>
<tr>
<td>103f8ef34d5c</td>
<td>9 hours ago</td>
<td>Merge with trunk</td>
</tr>
<tr>
<td>6ff565c008df</td>
<td>9 hours ago</td>
<td>[rest-extension] Fixed incorrectly formatted re-pair request on startup, fixes issue with sending</td>
</tr>
<tr>
<td>9a4ee307a9a1</td>
<td>9 hours ago</td>
<td>[rest-extension] Moved UnityScriptEditorSettings.json into the Library folder of the Unity project</td>
</tr>
<tr>
<td>ea7c770513d8</td>
<td>10 hours ago</td>
<td>Merge graphics/renderloop--lightsorting into trunk</td>
</tr>
<tr>
<td>642496b9d295</td>
<td>10 hours ago</td>
<td>[Windows Bootstrapper] Corrected paths in ini file and setting download size. Displaying de</td>
</tr>
<tr>
<td>6882dd3db094</td>
<td>10 hours ago</td>
<td>Fixed dynamic array test failures in ABV</td>
</tr>
<tr>
<td>64d11de876b0</td>
<td>11 hours ago</td>
<td>Removed 'title' bold text control and using the dialog's title instead, which is how apps are</td>
</tr>
<tr>
<td>9a2ea557c8ae</td>
<td>1 day ago</td>
<td>Improved wording of a few dialogs</td>
</tr>
</tbody>
</table>
Build system

• Modified JamPlus
  • “everything about it sucks, but it works”

• Both actual build & generates IDE projects

• Some rules are complex, e.g. script binding files that generate both C# and C++ code, etc.
Build/test farm

• Katana; fork of buildbot + custom frontend
  • Before: TeamCity
    • Before: Hudson (now Jenkins)
      • Before: crappy scripts coordinating two build machines via network share
        • Before: “Aras, can you build windows standalone and give it to me?”
<table>
<thead>
<tr>
<th>Builder</th>
<th>Current jobs</th>
<th>Last run</th>
<th>Status</th>
<th>Shortcuts Revision</th>
<th>Build Time</th>
<th>Run build</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABuildVerification [ABV]</td>
<td></td>
<td>2 hours ago (2 hours 31 mins, 15 secs)</td>
<td>#8546 Build Failed (dependency failed to build)</td>
<td>unity / 8d5e6091abf9</td>
<td>1m 29s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build AndroidDevelopmentPlayer [ABV]</td>
<td></td>
<td>4 hours ago (11 mins, 5 secs)</td>
<td>#9196 Build Finished Successfully</td>
<td>unity / 8d5e6091abf9</td>
<td>11m 5s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build AndroidPlayer [ABV]</td>
<td></td>
<td>4 hours ago (22 mins, 15 secs)</td>
<td>#9599 Build Finished Successfully</td>
<td>unity / 8d5e6091abf9</td>
<td>22m 15s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build BlackBerryPlayer [ABV]</td>
<td></td>
<td>4 hours ago (27 mins, 22 secs)</td>
<td>#9377 Build Finished Successfully</td>
<td>unity / 8d5e6091abf9</td>
<td>27m 21s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build BuiltinResources [Trunk-Nightly]</td>
<td></td>
<td>17 hours ago (1 hour 18 mins, 37 secs)</td>
<td>#812 Build Finished Successfully</td>
<td>unity / 642dd35406a4</td>
<td>20m 15s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build CacheServer</td>
<td></td>
<td>18 hours ago (3 mins, 21 secs)</td>
<td>#692 Build Finished Successfully</td>
<td>unity / 642dd35406a4</td>
<td>3m 20s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build EditorResources [Manual-Only]</td>
<td></td>
<td>13 days ago (55 mins, 45 secs)</td>
<td>#31 Build Finished Successfully</td>
<td>unity / 621cbb09e5b4</td>
<td>55m 44s</td>
<td>[View]</td>
</tr>
<tr>
<td>Build ExampleProject</td>
<td></td>
<td>17 hours ago (25 mins, 25 secs)</td>
<td>#798 Build Finished Successfully</td>
<td>unity / 642dd35406a4</td>
<td>12m 4s</td>
<td>[View]</td>
</tr>
</tbody>
</table>
Story 3
Typical work item
General work process

• Ideas/discussion
• Code
• Tests
• Review
• All above intermixed
e.g. “Let’s improve shadowmap filtering!”

• 2 hours of work, right?
Actual process #1

- Write up ideas
- Explain current shadow filtering to someone new
- Discuss possible approaches
Approach A

- Directly sample & filter shadow in the shaders
  - :) works on alpha blended
  - :) MSAA just works
  - :( DX9 SM3.0 might run into instruction limits
  - :( new filtering requires new set of shaders for everyone
  - :( wasted work due to overdraw / quad shading
  - :( cascade transitions
  - :( bias issues with large filter
Approach B

- Gather into screenspace, then blur
  :) isolated in two shaders
  :) can do fairly wide filter
  :( noisy
  :( depth discontinuity detection errors
  :( cache trashing
  :( MSAA edges
  :( alpha blended
Approach C

- Gather with filtering into screenspace
  - :D isolated in one shader
  - :( MSAA edges
  - :( alpha blended
  - :( cascade transitions
  - :( bias issues with large filter
Decision time!

- Current approach is B (gather to screen, then filter)
  - Done in 2007, driven by DX9 PS2.0 limits and no-hardware-PCF
    - Both aren’t a major concern today
- Let’s try C (gather&filter to screen)
  - Less invasive
  - No shader variant explosion
  - No DX9-level shader length issues
“Actual work” time

• Read “Shadow Mapping Summary” (The Witness) or “A Sampling of Shadow Techniques” (mynameismjp)
  • Knowing where to look is fairly useful!

• Write the shader

• Profit!
More work

• Let’s keep both during beta for a while, for user feedback
  • Means have to add settings & UI

• Add a graphics test, because that’s what we do
  • Turns out it does not work on Xbox 360 & Android GLES2.0
    • Aha! no hardware PCF there, and we forgot about that case
    • Write variant that does manual depth comparisons
      • Maybe we should have shadow sampling helper functions that deal with all this stuff
Compare performance

- Instruction count; AMD GPU ShaderAnalyzer; PVR Shader Editor
- Check performance on high/mid/low-end GPUs
  - Compare with old filtering approach
Code review

• Forgot to take shadow intensity into account
• No need to output depth, since no later blur pass
• DX9 SM2.0 fallback path
Push to production ("trunk")

- Get whole build/test suite to pass
- Someone else pushed built-in shaders build in the meantime; merge and rebuild
In total?

• Was probably 3-4 days

• Could be less, if were familiar with shadows stuff
  • But the guy who is familiar, was busy/away/…
And it’s not final yet!

• Will get feedback from beta
  • Could mean removing the old shadow filter
    • Which will change a lot of graphics test outputs, will need update
  • Better dealing with cascade transitions & bias issues

• Updating documentation
  • Update a bunch of screenshots
  • And translations
Story 4
You’re making stuff for others
Backwards compatibility

• Bad decisions are forever!
  • Or, almost

• Very tricky balance
  • “Everything keeps on working forever”
    • Old Microsoft. OpenGL.

• “Move fast and break things”
  • Facebook. Direct3D back in the day.
Sometimes replacing whole system with another is an option

• Will need to have the old one linger around for a while
  • Unless have a good way of auto-converting

• We did this with animation & particles; doing it with audio and partially lighting now.
Sometimes tools to upgrade are an option

- Scripting API breakage/obsoletion/renaming
  - Did a tool to change scripts both at source & CIL level

- Data format conversions
  - Mostly transparent, as long as behavior stays the same
Sometimes continued evolution is an option

• Add nicer parts, sweep nasty parts under the rug
  • Problems if nasty parts have taken all the nice API names already
Lots of thinking

- e.g. that “improve shadow map filtering” before
- Will change look of all games with shadows
- If 95% users like it but 5% don’t - is that okay?
  - What if 80% vs 20%?
  - 70% vs 30%?
Tools for others

- Their experience level
- Their use cases
- Their mental model of things
Tools for others

- Error messages from coders in 95% cases will confuse non-coders

- 10 numbers are confusing. Want a traffic light feedback instead
  - Except when need to dig in. Then need 100 numbers
Built-in vs extensible

• Having nice stuff built-in is nice
• Being able to change things is nice
• Being able to invent new usages within existing engine is nice
• A range from “Here’s a C compiler, do anything!” to “Here’s a button to make a game”
Interacting with users

• Tricky balance!
  • I could work on improving Unity
  • Or I could go and help Unity users

• Gets harder as
  • Number of users grow
  • Variance of users grow
  • Product & company grows
Story 5
Real world out there
Hardware landscape

• Very few people have NVIDIA Titans

• No one updates their drivers

• Most popular GPU? Intel GMA HD
  • Hey, a step up from Intel GMA950 / 945 at least

• Windows XP
  • Which means Direct3D 9
  • Why? China
Hardware range

- PC >10x performance range
- Mobiles >10x performance range
  - And insist on crazy resolutions
Real content

• Real scenes often aren’t teapots & bunnies

• More complexity & variety

• More “harder” cases
  • Non-manifold meshes, intersecting objects, holes, thin objects, depth precision, alpha blending, light setups, aliasing
Real content

- Stresses combinations & interplay of features
- Don’t know the worst case or ranges of parameters
“Creative” ways

• 4096 sprite or eyeball textures, without mipmaps

• Seven point lights in the same place, with shadows

• Insists on parsing XML at runtime

• Water that renders full scene for reflection at full LOD, with material that barely makes it visible

• Spends 90% of script time creating strings

• Turns on deferred rendering, for a 2D game with no lights
Creative ways

• OTOH, many will surprise you with really creative stuff

• Navigation in Monument Valley

• Space-scale physics & rendering in Kerbal Space Program

• ShaderForge serializing whole graph in the shader itself
So is engine coding boring?

• It can at times

• But also hugely satisfying to see it used in thousands of ways
  • By a few million people even
Q?